UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/664,055	09/17/2003	Geoff Barrett	1875.5100000	3239

26111 7590 04/05/2007

STERNE, KESSLER, GOLDSTEIN & FOX P.L.L.C.
1100 NEW YORK AVENUE, N.W.
WASHINGTON, DC 20005

EXAMINER	
LI, AIMEE J	
ART UNIT	**PAPER NUMBER**
2183	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	04/05/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	Application No.	Applicant(s)	
	10/664,055	BARRETT ET AL.	
	Examiner	Art Unit	
	Aimee J. Li	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *07 September 2006 and 08 January 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-2 and 4-24* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,2 and 4-24* is/are rejected.

7)☒ Claim(s) *23-24* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *07 September 2006* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-2 and 4-24 have been considered. Claim 3 has been cancelled as per Applicant's request. Claims 1,6-18, and 21 have been amended as per Applicant's request.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: Amendment as filed 07 September 2006; Drawings as filed 07 September 2006; Extension of Time for 1 Month as filed 07 September 2006; and Amendment as filed 08 January 2007.

Claim Objections

3. Claims 23-24 are objected to because of the following informalities: Please replace the language "the device of claim 8" and "the method of claim 1" with the actual text of claims 1 and 8. As is, it is unclear whether claims 23 and 24 are independent or dependent claims and the exact scope of the claim language due to its unclear reliance upon other claims. Appropriate correction is required.

Claim Rejections - 35 USC § 102

4. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

5. Claims 1-8 and 23-24 are rejected under 35 U.S.C. 102(e) as being taught by Paul et al., U.S. Patent Number 6,704,863 (herein referred to as Paul).

6. Referring to claims 1 and 24, taking claim 1 as exemplary, Paul has taught a method of processing an interrupt verification support mechanism in a computer system comprising a processor and an input for external interrupts communicatively coupled to the processor (Paul Abstract "A method, system and processor are provided for minimizing latency and loss of processor bandwidth in a pipelined processor when responding to an interrupt..."; column 1, lines 17-63 "...Many of these activities are driven by external events, which may occur randomly with respect to the sequence of operations being carried out by the CPU...An interrupt is a special type of input to the CPU. When an interrupt occurs..."; column 3, lines 31-55 "...A method is present herein for servicing and interrupt in a pipelined processor..."; column 4, lines 17-35 "...where the processor includes interrupt handling circuitry adapted to generate one or more interrupt-related instructions..."; and Figure 4), the method comprising the steps:

a. Processing at least one actual instruction in the processor (Paul column 5, lines 17-28 "...Typically, while one instruction executes in the final stage of such a pipeline, its successor is being decoded in the previous stage, and a third instruction is being fetched into the pipeline memory..."); and

b. If an external interrupt request or an interrupt pseudo-instruction is received by the processor (Paul column 7, lines 10-23 "...When an interrupt is received **70**..."; column 8, lines 26-64 "...The interrupt handling circuitry **100** for recognizing the DMA interrupt **104** and managing the pipeline..."; Figure 3; and Figure 5), replacing the actual instruction in an instruction fetch stage of the processor with the pseudo-instruction (Paul column 7, lines 10-23 "...the pipeline is stalled **76** at the insertion point and the first of the hardwired interrupt-related

instructions is inserted **78** into the pipeline..."; column 8, lines 26-64 "...When the interrupt handling circuitry detects that interrupt **104** is a BARq DMA request, it stalls the pipeline by sending a stall signal to fetch stage **86**, enables clocking of the hardwired instruction stack, and programs the multiplexer **96** to connect the hardwired instruction stack to the insertion point..."; Figure 3; and Figure 5 – In regards to Paul, when an interrupt is received, the current instruction in the fetch stage is stalled and the hardwired instructions are inserted into the decode, i.e. the current main program instruction in the fetch stage is replaced by the hardwired instructions in the decode stage via the multiplexer.).

7. Claim 24 is similar in limitations to claim 1 and is rejected for similar reasons.

8. Referring to claim 2, Paul has taught the method of claim 1 comprising processing at least one actual instruction in the processor in an instruction pipeline wherein instructions are processed concurrently by an instruction fetch stage (Paul column 8, lines 26-34 "...This example depicts a 3-stage pipeline in which instructions are fetched..." and Figure 5), an instruction decode stage (Paul column 8, lines 26-34 "...This example depicts a 3-stage pipeline in which instructions are...decoded..." and Figure 5), an instruction issue stage (Paul column 8, lines 26-34 "...A multiplexer **96** selects either the main program instruction...or the hardwired instruction..." and Figure 5 – In regards to Paul, the multiplexer is in essence an instruction issuance stage, since it determines which instruction, one from the main program or one hardwired for interrupts, to send to, i.e. issue, to the decode and execute stages.), an instruction execute stage (Paul column 8, lines 26-34 "...This example depicts a 3-stage pipeline in which instructions are...executed..." and Figure 5) and a result write-back stage (Paul Figure 1 and

Figure 6 – In regards to Paul, the term "result write-back stage" is never explicitly stated, but Paul's description of Figure 1 and Figure 6, as well as his description of the execute stage, infers that the instruction is retired, i.e. the result is written-back, from the pipeline. Paul further teaches that his system can work with any pipeline, i.e. a pipeline with the write-back stage separate from the execute stage, in column 9, lines 42-57 ("...Therefore, the method described herein is believed to be applicable to any pipelined processor...").).

9. Referring to claim 5, Paul has taught the method of claim 1 comprising simultaneously processing a number of instructions in the processor in an instruction pipeline with several instruction stages each instruction being in a different instruction stage at a time (Paul column 2, lines 1-34 "...in a pipelined processor, the steps are performed concurrently on multiple instructions, as they advance through the pipeline...").

10. Referring to claim 6, Paul has taught the method of claim 1 further comprising storing at least information of a program counter of the instruction which is to be interrupted and a sort of interrupt to use in a set of one or more interrupt registers of the processor (Paul column 7, lines 10-23 "...the pipeline is stalled **76** at the insertion point and the first of the hardwired interrupt-related instructions is inserted **78** into the pipeline..."; column 8, lines 26-64 "...When the interrupt handling circuitry detects that interrupt **104** is a BARq DMA request, it stalls the pipeline by sending a stall signal to fetch stage **86**, enables clocking of the hardwired instruction stack, and programs the multiplexer **96** to connect the hardwired instruction stack to the insertion point..."; Figure 3; Figure 5; and column 9, lines 25-41 "Other types of interrupts may be handled by the processor according to this method in the conventional way (i.e., when an interrupt occurs, the processor saves the current context [program counter and status] and

replaces the contents of the instruction pipeline with the beginning instructions of the interrupt service routine)..." – In regards to Paul, there are two types of interrupt handling taught in Paul: conventional handling and hardwire interrupt instruction insertion. Either way, the instruction address of the instruction being interrupted is stored, which is stored in the program counter during execution of the main program, and the cause of the interrupt is maintained somewhere. In the conventional system, the program counter and status are explicitly stored. In the hardwire, the program counter is not affected by the hardwired instructions, which are not fetched from memory but selected from a stack by a multiplexer, so the program counter is kept in the program counter. Both approaches use the available registers to execute the interrupt instructions as well as the main program instructions. Hence, the system's program counter and registers are also interrupt registers, since they are used during interrupt processing.).

11. Referring to claim 7, Paul has taught the method of claim 1 further comprising comparing data content of a program counter with data content of an interrupt register and replacing the actual instruction with a pseudo-instruction when the data content of the program counter matches the data content of the interrupt register, or when an external interrupt is present (Paul column 7, lines 10-23 "...the pipeline is stalled **76** at the insertion point and the first of the hardwired interrupt-related instructions is inserted **78** into the pipeline..."; column 8, lines 26-64 "...When the interrupt handling circuitry detects that interrupt **104** is a BARq DMA request, it stalls the pipeline by sending a stall signal to fetch stage **86**, enables clocking of the hardwired instruction stack, and programs the multiplexer **96** to connect the hardwired instruction stack to the insertion point..."; Figure 3; Figure 5; and column 9, lines 25-41 "Other types of interrupts may be handled by the processor according to this method in the conventional way (i.e., when an

interrupt occurs, the processor saves the current context [program counter and status] and replaces the contents of the instruction pipeline with the beginning instructions of the interrupt service routine)...").

12. Referring to claims 8 and 23, taking claim 23 as exemplary, Paul has taught an interrupt verification support mechanism device for a computer system comprising

a. A processor and an input for external interrupt requests or interrupt pseudo-instructions communicatively coupled to the processor (Paul Abstract "A method, system and processor are provided for minimizing latency and loss of processor bandwidth in a pipelined processor when responding to an interrupt..."; column 1, lines 17-63 "...Many of these activities are driven by external events, which may occur randomly with respect to the sequence of operations being carried out by the CPU...An interrupt is a special type of input to the CPU. When an interrupt occurs..."; column 3, lines 31-55 "...A method is present herein for servicing and interrupt in a pipelined processor..."; column 4, lines 17-35 "...where the processor includes interrupt handling circuitry adapted to generate one or more interrupt-related instructions..."; and Figure 4),

b. Wherein the device includes a set of one or more interrupt registers each of which contains information, the information including at least a program counter of the instruction which is to be interrupted and a sort of interrupt to use (Paul column 7, lines 10-23 "...the pipeline is stalled **76** at the insertion point and the first of the hardwired interrupt-related instructions is inserted **78** into the pipeline..."; column 8, lines 26-64 "...When the interrupt handling circuitry detects that

interrupt **104** is a BARq DMA request, it stalls the pipeline by sending a stall signal to fetch stage **86**, enables clocking of the hardwired instruction stack, and programs the multiplexer **96** to connect the hardwired instruction stack to the insertion point..."; Figure 3; Figure 5; and column 9, lines 25-41 "Other types of interrupts may be handled by the processor according to this method in the conventional way (i.e., when an interrupt occurs, the processor saves the current context [program counter and status] and replaces the contents of the instruction pipeline with the beginning instructions of the interrupt service routine)..." – In regards to Paul, there are two types of interrupt handling taught in Paul: conventional handling and hardwire interrupt instruction insertion. Either way, the instruction address of the instruction being interrupted is stored, which is stored in the program counter during execution of the main program, and the cause of the interrupt is maintained somewhere. In the conventional system, the program counter and status are explicitly stored. In the hardwire, the program counter is not affected by the hardwired instructions, which are not fetched from memory but selected from a stack by a multiplexer, so the program counter is kept in the program counter. Both approaches use the available registers to execute the interrupt instructions as well as the main program instructions. Hence, the system's program counter and registers are also interrupt registers, since they are used during interrupt processing.), so as to enable the device to process at least one actual instruction (Paul column 5, lines 17-28 "...Typically, while one instruction executes in the final stage of such a pipeline, its successor is being

decoded in the previous stage, and a third instruction is being fetched into the pipeline..."), and

c. If an external interrupt request is received by the processor (Paul column 7, lines 10-23 "...When an interrupt is received **70**..."; column 8, lines 26-64 "...The interrupt handling circuitry **100** for recognizing the DMA interrupt **104** and managing the pipeline..."; Figure 3; and Figure 5), the at least one actual instruction is replaced with the pseudo-instruction (Paul column 7, lines 10-23 "...the pipeline is stalled **76** at the insertion point and the first of the hardwired interrupt-related instructions is inserted **78** into the pipeline..."; column 8, lines 26-64 "...When the interrupt handling circuitry detects that interrupt **104** is a BARq DMA request, it stalls the pipeline by sending a stall signal to fetch stage **86**, enables clocking of the hardwired instruction stack, and programs the multiplexer **96** to connect the hardwired instruction stack to the insertion point..."; Figure 3; and Figure 5 – In regards to Paul, when an interrupt is received, the current instruction in the fetch stage is stalled and the hardwired instructions are inserted into the decode, i.e. the current main program instruction in the fetch stage is replaced by the hardwired instructions in the decode stage via the multiplexer.).

13. Claim 23 contains similar limitations to claim 8 and is rejected for similar reasons.

Claim Rejections - 35 USC § 103

14. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

15. Claims 4 and 20-22 are rejected under 35 U.S.C. 103(a) as being unpatentable over Paul et al., U.S. Patent Number 6,704,863 (herein referred to as Paul) as applied to claims 1 and 8 above, and further in view of Sproul, III, U.S. Patent Number 4,498,136 (herein referred to as Sproul).

16. Referring to claim 4, Paul has not taught creating the pseudo-instruction by a co-processor connected to the processor. Sproul has taught creating the pseudo-instruction by a co-processor connected to the processor (Sproul Abstract; column 4, lines 16-37; and Figure 2). In regards to Sproul, the device disclosed is a separate interrupt processor to assist with interrupt decoding and handling. A person of ordinary skill in the art at the time the invention was made, and as taught by Sproul, would have recognized that the interrupt co-processor of Sproul improves interrupt management in a pipeline (Sproul column 3, lines 10-23). Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the interrupt co-processor methods of Sproul in the device of Case to improve interrupt handling in a pipelined system.

17. Referring to claim 20, Paul in view of Sproul has taught wherein the pseudo-instruction is created by a co-processor connected to the processor (Sproul Abstract; column 4, lines 16-37; and Figure 2). In regards to Sproul, the device disclosed is a separate interrupt processor to assist with interrupt decoding and handling.

18. Referring to claim 21, Paul in view of Sproul has taught wherein the device is a media decoding system, the processor is a core decoder processor (Paul column 5, lines 17-25 "...the

processor may operate in a graphics-intensive application..."; column 8, lines 26-64 "...a 3-stage pipeline in which instructions are fetched **86**, decoded **88**..."; and Figure 5)) and the co-processor is a decoding accelerator adapted to assist the core processor with a decoding function (Sproul Abstract; column 4, lines 16-37; and Figure 2).

19. Referring to claim 22, Paul in view of Sproul has taught the device of claim 20 wherein the processor is a reduced instruction set computer (RISC) processor (Paul column 3, lines 31-39 "...the method applies to a pipelined processor having a RISC architecture...").

20. Claims 9-11 and 13-19 are rejected under 35 U.S.C. 103(a) as being unpatentable over Paul et al., U.S. Patent Number 6,704,863 (herein referred to as Paul) as applied to claim 8 above, and further in view of Case et al., U.S. Patent Number 4,777,587 (herein referred to as Case).

21. Referring to claim 9, Paul has taught the device of claim 8 wherein the device further comprises

a. An instruction fetch with a program counter and an interrupt register (Paul column 7, lines 10-23 "...the pipeline is stalled **76** at the insertion point and the first of the hardwired interrupt-related instructions is inserted **78** into the pipeline..."; column 8, lines 26-64 "...When the interrupt handling circuitry detects that interrupt **104** is a BARq DMA request, it stalls the pipeline by sending a stall signal to fetch stage **86**, enables clocking of the hardwired instruction stack, and programs the multiplexer **96** to connect the hardwired instruction stack to the insertion point..."; Figure 3; Figure 5; and column 9, lines 25-41 "Other types of interrupts may be handled by the processor according to

this method in the conventional way (i.e., when an interrupt occurs, the processor saves the current context [program counter and status] and replaces the contents of the instruction pipeline with the beginning instructions of the interrupt service routine)..."),

b. The instruction fetch being coupled to a first input of a multiplexer for transmitting instructions to said multiplexer (Paul column 8, lines 26-64 "...A multiplexer **96**..." and Figure 5), and

c. A second input of the multiplexer connected to an interrupt pseudo-instruction input (Paul column 8, lines 26-64 "...A multiplexer **96**..." and Figure 5).

22. Paul has not taught and the program counter connected with the interrupt register by a comparator. Case has taught the program counter connected with the interrupt register by a comparator (Case column 1, lines 42-50; column 2, lines 47-59; column 2, line 26 to column 3, line 63; Figure 1; column 5, line 24 to column 6, line 44; and Figures 3A and 3B). In regards to Paul, while he focuses on a high-speed execution of low-latency interrupts, he does teach that a conventional interrupt system is needed within his system (Paul Figure 3). Case teaches a conventional interrupt handling system. A person of ordinary skill in the art at the time the invention was made, and as taught by Case, would have recognized that performing branches in a single cycle while providing accurate branch prediction (Case column 1, lines 42-50) reduces the amount of time needed to branch to an interrupt handler, thereby improving the performance of a computer system.

23. Referring to claim 10, Paul in view of Case has taught The device of claim 9 wherein the second input of the multiplexer is capable of receiving interrupt pseudo-instruction signals or external interrupt requests (Paul column 8, lines 26-64 "...A multiplexer **96**..." and Figure 5).

24. Referring to claim 11, Paul in view of Case has taught wherein the comparator creates a high level signal only if the data content of the program counter matches the data content of the interrupt register (Case column 1, lines 42-50; column 2, lines 47-59; column 2, line 26 to column 3, line 63; Figure 1; column 5, line 24 to column 6, line 44; and Figures 3A and 3B).

25. Referring to claim 13, Paul in view of Case has taught wherein, when the data content of the program counter matches the data content of the interrupt register, the actual instruction is replaced with a pseudo-instruction (Case column 1, lines 42-50; column 2, lines 47-59; Figure 1; column 5, line 24 to column 6, line 44; and Figures 3A and 3B).

26. Referring to claim 14, Paul in view of Case has taught the device of claim 9 wherein when an external interrupt request is present at the multiplexer, the actual instruction is replaced with an interrupt pseudo-instruction (Paul column 7, lines 10-23 "...the pipeline is stalled **76** at the insertion point and the first of the hardwired interrupt-related instructions is inserted **78** into the pipeline..."; column 8, lines 26-64 "...When the interrupt handling circuitry detects that interrupt **104** is a BARq DMA request, it stalls the pipeline by sending a stall signal to fetch stage **86**, enables clocking of the hardwired instruction stack, and programs the multiplexer **96** to connect the hardwired instruction stack to the insertion point..."; Figure 3; and Figure 5 – In regards to Paul, when an interrupt is received, the current instruction in the fetch stage is stalled and the hardwired instructions are inserted into the decode, i.e. the current main program

instruction in the fetch stage is replaced by the hardwired instructions in the decode stage via the multiplexer.).

27. Referring to claim 15, Paul in view of Case has taught the device of claim 9 wherein an instruction coming from an output of the multiplexer is sequentially processed in an instruction pipeline of the processor (Paul column 8, lines 26-64 "...A multiplexer **96**..." and Figure 5).

28. Referring to claim 16, Paul in view of Case has taught the device of claim 9 wherein an instruction pipeline of the processor includes an instruction fetch stage (Paul column 8, lines 26-34 "...This example depicts a 3-stage pipeline in which instructions are fetched..." and Figure 5), an instruction decode stage (Paul column 8, lines 26-34 "...This example depicts a 3-stage pipeline in which instructions are...decoded..." and Figure 5), an instruction issue stage (Paul column 8, lines 26-34 "...A multiplexer **96** selects either the main program instruction...or the hardwired instruction..." and Figure 5 – In regards to Paul, the multiplexer is in essence an instruction issuance stage, since it determines which instruction, one from the main program or one hardwired for interrupts, to send to, i.e. issue, to the decode and execute stages.), an instruction execute stage (Paul column 8, lines 26-34 "...This example depicts a 3-stage pipeline in which instructions are...executed..." and Figure 5) and a result write-back stage (Paul Figure 1 and Figure 6 – In regards to Paul, the term "result write-back stage" is never explicitly stated, but Paul's description of Figure 1 and Figure 6, as well as his description of the execute stage, infers that the instruction is retired, i.e. the result is written-back, from the pipeline. Paul further teaches that his system can work with any pipeline, i.e. a pipeline with the write-back stage separate from the execute stage, in column 9, lines 42-57 ("...Therefore, the method described herein is believed to be applicable to any pipelined processor...").).

29. Referring to claim 17, Paul in view of Case has taught wherein the interrupt pseudo-instruction effects the instruction state stages required by the interrupt (Case column 1, lines 42-50; column 2, lines 47-59; column 2, line 26 to column 3, line 63; Figure 1; column 5, line 24 to column 6, line 44; and Figures 3A and 3B).

30. Referring to claim 18, Paul in view of Case has taught wherein if an interrupt request or an interrupt pseudo-instruction is received by the processor, the processor is adapted to cancel an instruction that is in the instruction fetch stage when the interrupt request or the interrupt pseudo-instruction is received and to reissue the instruction starting at the instruction fetch stage (Case column 1, lines 42-50; column 2, lines 47-59; column 2, line 26 to column 3, line 63; Figure 1; column 5, line 24 to column 6, line 44; and Figures 3A and 3B).

31. Referring to claim 19, Paul in view of Case has taught wherein if an interrupt request or an interrupt pseudo-instruction is received by the processor, the processor is adapted to cancel an instruction that is in any instruction stage when the interrupt request or the interrupt pseudo-instruction is received and to reissue the instruction starting at the instruction fetch stage (Case column 1, lines 42-50; column 2, lines 47-59; column 2, line 26 to column 3, line 63; Figure 1; column 5, line 24 to column 6, line 44; and Figures 3A and 3B).

32. Claim 12 is rejected under 35 U.S.C. 103(a) as being unpatentable over Paul et al., U.S. Patent Number 6,704,863 (herein referred to as Paul) further in view of Case et al., U.S. Patent Number 4,777,587 (herein referred to as Case) as applied to claim 9 above, and further in view of Sproul, III, U.S. Patent Number 4,498,136 (herein referred to as Sproul). Paul in view of Case has not taught wherein the output of the comparator is connected to the first input of an or-operator and the second input of the or-operator is connected to an interrupt controller so as to

enable the or-operator to create a high level signal if the signal received from the interrupt controller differs from the signal received from the comparator. Sproul has taught wherein the output of the comparator is connected to the first input of an or-operator and the second input of the or-operator is connected to an interrupt controller so as to enable the or-operator to create a high level signal if the signal received from the interrupt controller differs from the signal received from the comparator (Sproul Abstract; column 4, lines 16-37; and Figure 2). A person of ordinary skill in the art at the time the invention was made, and as taught by Sproul, would have recognized that the interrupt co-processor of Sproul improves interrupt management in a pipeline (Sproul column 3, lines 10-23). Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the interrupt co-processor methods of Sproul in the device of Case to improve interrupt handling in a pipelined system.

Response to Arguments

33. Examiner withdraws the objection to the specification in favor of the amended specification and replacement drawings.

34. Examiner withdraws the objection to the drawings in favor of the amended specification and replacement drawings.

35. Examiner withdraws rejections under 35 U.S.C. § 112, second paragraph in favor of the amended claims and in view of the arguments presented on these rejections.

36. Applicant's arguments with respect to the prior art rejections of claims 1-3 and 5-24 have been considered but are moot in view of the new ground(s) of rejection.

Conclusion

37. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

38. A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

39. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Aimee J. Li whose telephone number is (571) 272-4169. The examiner can normally be reached on M-T 7:00am-4:30pm.

40. If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

41. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Aimee J Li
Examiner
Art Unit 2183

1 April 2007